

Compositional Semantics in Verbmobil

Johan Bos Björn Gambäck Christian Lieske
Yoshiki Mori Manfred Pinkal Karsten Worm

Department of Computational Linguistics
University of the Saarland
Postfach 151150
D-66041 Saarbrücken, Germany*
e-mail: vm@coli.uni-sb.de

To appear in the Proceedings of COLING '96

Abstract

The paper discusses how compositional semantics is implemented in the Verbmobil speech-to-speech translation system using LUD, a description language for underspecified discourse representation structures. The description language and its formal interpretation in DRT are described as well as its implementation together with the architecture of the system's entire syntactic-semantic processing module. We show that a linguistically sound theory and formalism can be properly implemented in a system with (near) real-time requirements.

1 Introduction

Contemporary syntactic theories are normally unification-based and commonly aim at specifying as much as possible of the peculiarities of specific language constructions in the lexicon rather than in the “traditional” grammar rules. When doing semantic interpretation within such a framework, we want a formalism which allows for

- compositionality,
- monotonicity, and
- underspecification.

Compositionality may be defined rather strictly so that the interpretation of a phrase always should be the (logical) sum of the interpretations of its subphrases. A semantic formalism being compositional in this strict sense would also trivially be *monotonic*, since no destructive changes would need to be undertaken while building the

interpretation of a phrase from those of its subphrases.¹

However, compositionality is more commonly defined in a wider sense, allowing for other mappings from subphrase-to-phrase interpretation than the sum, as long as the mappings are such that the interpretation of the phrase still is a function of the interpretations of the subphrases. A common such mapping is to let the interpretation of the phrase be the interpretation of its (semantic) head modified by the interpretations of the adjuncts. If this modification is done by proper unification, the monotonicity of the formalism will still be guaranteed.

In many applications for Computational Linguistics, for example when doing semantically based translation — as in Verbmobil, the German national spoken language translation project described in Section 2 — a complete interpretation of an utterance is not always needed or even desirable. Instead of trying to resolve ambiguities, for example the ones introduced by different possible scopings of quantifiers, the interpretation of the ambiguous part is left unresolved. The semantic formalism of such a system should thus allow for the *underspecification* of these unresolved ambiguities (but still allow for them to be resolved in a monotonic way, of course). An underspecified form representing an utterance is then the representation of a *set* of meanings, all the possible interpretations of the utterance.

The rest of the paper is structured as follows. Section 2 gives an overview of the Verbmobil Project. Section 3 introduces LUD (description Language for Underspecified Discourse representations), the semantic formalism we use. Section 4 compares our approach to that of others for similar tasks. The actual implementation is described

*This research was funded by the German Federal Ministry of Education, Science, Research, and Technology (BMBF) under grant number 01 IV 101 R.

¹More formally, a semantic representation is monotonic iff the interpretation of a category on the right side of a rule subsumes the interpretation of the left side of the rule.

in Section 5, which also discusses coverage and points to some areas of further research. Finally, Section 6 sums up the previous discussion.

2 The Verbmobil Project

The project Verbmobil funded by the German Federal Ministry of Research and Technology (BMBF) combines speech technology with machine translation techniques in order to develop a system for translation in face-to-face dialogues. The overall project is described in (Wahlster, 1993); in this section we will give a short overview of the key aspects.

The ambitious overall objective of the Verbmobil project is to produce a device which will provide English translations of dialogues between German and Japanese businessmen who only have a restricted active, but larger passive knowledge of English. The domain is the scheduling of business appointments. The major requirement is to provide translations as and when users need them, and do so robustly and in (near) real-time.

In order to achieve this, the system is composed of time-limited processing components which on the source language (German or Japanese) side perform speech recognition, syntactic, semantic and pragmatic analysis, as well as dialogue management; transfer on a semantic level; and on the target language (English) side generation and speech synthesis. When the users speak English, only keyword spotting for the dialogue management is undertaken.

At any moment in the dialogue, a user may activate the Verbmobil device and start speaking his/her native language. The speech recognition component then processes the input and produces a word lattice representing the speech hypotheses and their corresponding prosodic information. The parsing component processes the lattice and assigns each well-formed path through it one or several syntactic and (compositional) semantic representations. Ambiguities introduced by these may be resolved by a resolution component. The representations produced are then assigned dialogue acts and used to update the model of the discourse, which in turn may be used by the speech recognizer to choose the current language model. The transfer component takes the (possibly resolved) semantic analysis of the input and builds a target language representation. The generator then constructs the corresponding English expression. For robustness, this deep-level processing strategy is complemented with a shallow analysis-and-transfer component.

3 Underspecified Representations

3.1 Theoretical Background

Since the Verbmobil domain is related to discourse rather than isolated sentences, a variant of Kamp's Discourse Representation Theory, DRT (Kamp and Reyle, 1993) has been chosen as the model theoretic semantics. However, to allow for underspecification of several linguistic phenomena, we have chosen a formalism that is suited to represent underspecified structures: LUD, a description language for underspecified discourse representations (Bos, 1995). The basic idea is the one given in Section 1, namely that natural language expressions are not directly translated into Discourse Representation Structures (DRSs), but into a representation that describes *several* DRSs.

Representations in LUD have the following distinct features. Firstly, all elementary semantic "bits" (conditions, entities, and events) are uniquely labeled. This makes them easy to refer to and results in a very powerful description language. Secondly, meta variables over DRSs (which we call *holes*) allow for the assignment of underspecified scope to a semantic operator. Thirdly, a subordination relation on the set of holes and labels constrains the number of interpretations of the LUD-representation in the object language: DRSs.

3.2 LUD-Representations

A LUD-representation U is a triple

$$\langle H_U, L_U, C_U \rangle$$

where H_U is a set of holes (variables over labels), L_U is a set of labeled (LUD) conditions, and C_U is a set of constraints. A *plugging* is a bijective function from holes to labels. For each plugging there is a corresponding DRS. The syntax of LUD-conditions is formally defined as follows:

1. If x is a discourse marker (i.e., entity or event), then $dm(x)$ is a LUD-condition;
2. If R is a symbol for an n -place relation, x_1, \dots, x_n are discourse markers, then $pred(R, x_1, \dots, x_n)$ is a LUD-condition;
3. If l is a label or hole for a LUD-condition, then $\neg l$ is a LUD-condition;
4. If l_1 and l_2 are labels (or holes) for LUD-conditions, then $l_1 \rightarrow l_2$, $l_1 \wedge l_2$ and $l_1 \vee l_2$ are LUD-conditions;
5. Nothing else is a LUD-condition.

There are three types of constraints in LUD-representations. There is *subordination* (\leq), *strict subordination* ($<$), and finally *presupposition* (α). These constraints are syntactically defined as:

If l_1, l_2 are labels, h is a hole, then $l_1 \leq h$,
 $l_1 < l_2$ and $l_1 \alpha l_2$ are LUD-constraints.

The interpretation of a LUD-representation is the interpretation of **top**, the label or hole of a LUD-representation for which there exists no label that subordinates it.²

The interpretation function I is a function from a labeled condition to a DRS. This function is defined with respect to a plugging P . We represent a DRS as a box $\boxed{D \mid C}$, where D is the set of discourse markers and C is the set of conditions. The mappings between LUD-conditions and DRSs are then defined in (2)-(9) where l is a label or hole and ϕ is a labeled condition.

$$I_P(l) = \begin{cases} I(\phi) \text{ iff } l : \phi \in L_U \end{cases} \quad (1)$$

$$I_P(l) = \begin{cases} I(P(l)) \text{ iff } l \in H_U \end{cases} \quad (2)$$

$$I(dm(x)) = \left\{ \boxed{x \mid} \right\} \quad (3)$$

$$I(pred(R, x_1, \dots, x_n)) = \left\{ \boxed{\mid R(x_1, \dots, x_n)} \right\} \quad (4)$$

$$I(l_1 \wedge l_2) = \{ K_1 \otimes K_2 \mid K_1 \in I(l_1) \ \& \ K_2 \in I(l_2) \} \quad (5)$$

$$I(l_1 \rightarrow l_2) = \left\{ \boxed{\mid K_1 \rightarrow K_2} \mid K_1 \in I(l_1) \ \& \ K_2 \in I(l_2) \right\} \quad (6)$$

$$I(l_1 \vee l_2) = \left\{ \boxed{\mid K_1 \vee K_2} \mid K_1 \in I(l_1) \ \& \ K_2 \in I(l_2) \right\} \quad (7)$$

$$I(\neg l_1) = \left\{ \boxed{\mid \neg K_1} \mid K_1 \in I(l_1) \right\} \quad (8)$$

In (6) \otimes is the merge operation, that takes two DRSs K_1 and K_2 and returns a DRS which domain is the union of the set of the domains of K_1 and K_2 , and which conditions form the union of the set of the conditions of K_1 and K_2 .

²The reader interested in a more detailed discussion of the interpretation of underspecified semantic representations is referred to (Bos, 1995).

3.3 Lexical Entries and Composition

For building LUD-representations we use a lambda-operator and functional application in order to compositionally combine simple LUD-representations to complex ones. In addition, we have two functions that help us to keep track of the right labels. These are **top**, as described above, and **main**, the label of the semantic head of a LUD-representation. Further, we have an operation that combines two LUD-representations into one: \oplus (merge for LUD-representations). Some sample lexical entries for German

4 Related Work

The LUD representation is quite closely related to UDRSs, underspecified DRSs (Reyle, 1993). The main difference is that the LUD description language in principle is independent of the object language, thus not only DRT, but also ordinary predicate logic, as well as a Dynamic Predicate Logic (Groenendijk and Stokhof, 1991) can be used as the object language of LUD, as shown in (Bos, 1995). Compared to UDRS, LUD also has a stronger descriptive power: Not DRSs, but the smallest possible semantic components are uniquely labeled.

The Verbmobil system is a translation system built by some 30 different groups in three countries. The semantic formalism used on the English generation side has been developed by CSLI, Stanford and is called MRS, Minimal Recursion Semantics (Copestake et al., 1995). The deep-level syntactic and semantic German processing of Verbmobil is also done along two parallel paths. The other path is developed by IBM, Heidelberg and uses a variant of MRS, Underspecified Minimal Recursion Semantics (UMRS) (Egg and Lebeth, 1995). All the three formalisms LUD, MRS, and UMRS have in common that they use a flat, neo-Davidsonian representation and allow for the underspecification of functor-argument relations. In MRS, this is done by unification of the relations with unresolved dependencies. This, however, results in structures which cannot be further resolved. In UMRS this is modified by expressing the scoping possibilities directly as disjunctions. The main difference between both types of MRSs and LUD is that the interpretation of LUD in an object language other than ordinary predicate logic is well defined, as described in Section 3.2.

The translation task of the SICS-SRI Bilingual Conversation Interpreter, BCI (Alshawi et al., 1991) is quite similar to that of Verbmobil. The BCI does translation at the level of Quasi-Logical Form, QLF which also is a monotonic

$$\begin{aligned}
\text{das} : \quad & \lambda P. < \{ \quad \}, \{ l_i : dm(z) \}, \{ l_i \alpha \mathbf{main}(P) \} > \oplus P(z) \\
\text{geht} : \quad & \lambda y. \lambda e. < \{ h_l \}, \left\{ \begin{array}{l} l_i : pred(gehen, e), \\ l_j : pred(theme, e, y), \\ l_k : l_i \wedge l_j \end{array} \right\}, \{ l_k \leq h_l \} > \\
\text{jeder} : \quad & \lambda P. \lambda Q. < \{ h_i \}, \left\{ \begin{array}{l} l_j : dm(x), \\ l_k : l_j \wedge \mathbf{main}(P), \\ l_l : l_k \rightarrow h_i \end{array} \right\}, \left\{ \begin{array}{l} l_l \leq \mathbf{top}(Q), \\ \mathbf{main}(Q) \leq h_i \end{array} \right\} > \oplus P(z) \oplus Q(z) \\
\text{termin} : \quad & \lambda x. < \{ \quad \}, \{ l_i : termin(x) \}, \{ \quad \} > \\
\text{das geht} : \quad & \lambda e. < \{ h_0 \}, \left\{ \begin{array}{l} l_4 : dm(z), \\ l_5 : pred(gehen, e), \\ l_6 : pred(theme, e, z), \\ l_7 : l_5 \wedge l_6 \end{array} \right\}, \left\{ \begin{array}{l} l_7 \leq h_0, \\ l_4 \alpha_i l_7 \end{array} \right\} >
\end{aligned}$$

Figure 1: Lexical entries and a sample derivation in LUD

representation language for compositional semantics as discussed in (Alshawhi and Crouch, 1992). The QLF formalism incorporates a Davidsonian approach to semantics, containing underspecified quantifiers and operators, as well as ‘anaphoric terms’ which stand for entities and relations to be determined by reference resolution. In these respects, the basic ideas of the QLF formalism are quite similar to LUD.

5 Syntax–Semantics Interface and Implementation

5.1 Grammar

The LUD semantic construction component has been implemented in the grammar formalism TUG, Trace and Unification Grammar (Block and Schachtl, 1992), in a system called TrUG (in cooperation with Siemens AG, Munich, who provided the German syntax and the TrUG system). TUG is a formalism that combines ideas from Government and Binding theory, namely the use of traces, with unification in order to account for, for example, the free word order phenomena found in German.

5.1.1 Syntax and Semantics

A TUG grammar basically consists of PATR-II style context free rules with feature annotations. Each syntactic rule gets annotated with a semantic counterpart. In this way, syntactic derivation and semantic construction are fully interleaved and semantics can further constrain the possible readings of the input.

In order to make our formalisation executable,

we employ the TrUG system, which compiles our rules into an efficient Tomita-style parser. In addition TrUG incorporates sortal information, which is used to rank parsing results.

Consider a simplified example of a syntactic rule annotated with a semantic functor–argument application.

```

s ----> np, vp |
      np:agr = vp:agr,
      lud_fun_arg(s, vp, np) .

```

In this example, a sentence **s** consists of an **np** and a **vp**. The first feature equation annotated to this rule says that the value of the feature **agr** (for agreement) of the **np** equals that of the respective feature value of the **vp**.

5.1.2 The Composition Process

A category symbol like **np** in the rule above also stands for the entry node of its associated feature structure. This property is used for the semantic counterpart of the rule: **lud_fun_arg** is a call to a semantic rule, a *macro* in the TUG notation, which defines functor–argument application. Since the macro gets the entry nodes of the feature structures as arguments, all the information present in the feature structures can be accessed within the macro which is defined as

```

lud_fun_arg(Result, Fun, Arg) =>
  lud_context_equal(Fun, Result),
  context(Fun, FunContext),
  context(Arg, ArgContext),
  subcat(Result, ResultSc),
  subcat(Fun, [ArgContext | ResultSc]) .

```

The functor–argument application is based on the notion of the *context* of a LUD-representation. The context of a LUD-representation is a three-place structure consisting of the LUD-representation’s main label and top hole (as described in Section 3.3) and its main instance, which is a discourse marker or a lambda-bound variable. A LUD-representation also has a semantic subcategorization list under the feature **subcat** which performs the same function as a λ -prefix. This list consists of the contexts of the arguments a category is looking for.

The functor–argument application macro thus says the following. The context of the result is the context of the functor. The functor is looking for the argument as the first element on its **subcat** list, while the result’s **subcat** list is that of the functor minus the argument (which has been bound in the rule). The binding of variables between functor and argument takes place via the **subcat** list, through which a functor can access the main instance and the main label of its arguments and state relations between them.

Note that the only relevant piece of information contained in a LUD-representation for the purpose of composition is its context. Its content in terms of semantic predicates is handled differently. The predicates of a LUD-representation are stored in a special slot provided for each category by the TrUG system. The contents of this slot is handed up the tree from the daughters to the mother completely monotonically. So the predicates introduced by some lexical entry percolate up to the topmost node automatically.

These two restrictions, the use of only a LUD-representation’s context in composition and the monotonic percolation of semantic predicates up the tree, make the system completely compositional in the sense defined in Section 1.

5.1.3 The lexicon

To see how the composition interacts with the lexicon, consider the following lexical macro defining the semantics of a transitive verb

```
trans_verb_sem(Cat,Rel,[Role1,Role2]) =>
  basic_pred(Rel,Inst,L1),
  udef(Inst,L2),
  group([L1,L2,ArgL1,ArgL2],Main),
  leq(Main,Top),
  lud_context(Cat,Inst,Main,Top).
role(Inst,Role1,Arg1,ArgL1),
role(Inst,Role2,Arg2,ArgL2),
subcat(Cat,[lud(Arg1,_,_),
            lud(Arg2,_,_)])
```

The macro states that a transitive verb introduces a basic predicate of a certain relation with an instance and a label. The instance is related to its two arguments by argument roles. The arguments’ instances are accessed via the verb’s **subcat** list (and get bound during functor–argument application, cf. above). The labels introduced are grouped together; the group label is the main label of the LUD-representation, the instance its main instance. Another property of the verb’s semantics is that it introduces the top hole of the sentence.

5.2 Interfaces to Other Components

As sketched in Section 2, our semantic construction component delivers output to the components for semantic evaluation and transfer. The paragraphs that follow describe the common interface to these two components.

5.2.1 Resolution of Underspecification

Generating a scopally resolved LUD-representation from an underspecified one is the process which we referred to as plugging in Section 3.2. It aims at making the possibly ambiguous semantics captured by a LUD unique. Obviously, purely mathematical approaches for transforming the partial ordering encoded in the **leq** constraints into a total ordering may yield many results.

Fortunately, linguistic constraints allow us to reduce the effort that has to be put into the computation of pluggings. An example is the linguistic observation that a predicate that encodes sentence mood in many cases modifies all of the remainder of the proposition for a sentence. Thus, pluggings where the predicate for sentence mood is subject to a **leq** constraint should not be considered. They would result in a resolved structure expressing that the mood-predicate does not have scope over the remaining proposition. This would be contrary to the linguistic observation.

5.2.2 Supplementary Information

As a supplement to semantic predicates, our output contains various kinds of additional information. This is caused by the overall architecture of the Verbmobil system which does not provide for fully-interconnected components. There is, e.g., no direct connection

between the speech recognizer and the component for semantic evaluation. Thus, our component has to pipe certain kinds of information (like prosodic values). Accordingly, our output consists of “Verbmobil Interface Terms” (VITs), which differ slightly from the LUD-terms described above

mainly in that they include non-semantic information.

5.3 Implementation Status

Currently, the lexicon of the implemented system contains about 1400 entries (full forms) and the grammar consists of about 400 syntactic rules, of which about 200 constitute a subgrammar for temporal expressions. The system has been tested on three simplified dialogues from a corpus of spoken language appointment scheduling dialogues collected for the project and processes about 90% of the turns the syntax can deal with.

The system is currently being extended to cover nine additional dialogues from the corpus completely. The size of the lexicon will then be about 2500 entries, which amounts to about 1700 lemmata.

6 Conclusions

We have discussed the implementation of a compositional semantics in the Verbmobil speech-to-speech translation system. The notions of monotonicity and underspecification were discussed and LUD, a description language for underspecified discourse representation structures was introduced. As shown in Section 3, the LUD description language has a well-defined interpretation in DRT. Differently from Reyle's UDRSs, however, LUD assigns labels to the minimal semantic element and may also be interpreted in other object languages than DRT.

The key part of the paper, Section 5, showed how the linguistically sound LUD formalism has been properly implemented in a (near) real-time system. The implementation in Siemens' TUG grammar formalism was described together with the architecture of the entire semantic processing module of Verbmobil and its current coverage.

7 Acknowledgements

We are gratefully indebted to Scott McGlashan and CJ Rupp who both worked on parts of the implementation. The results of the paper have greatly benefitted from the cooperation with our other colleagues in Verbmobil, especially those at IBM and CSLI, as well as the ones working on the modules closest to ours in the processing chain.

A number of people have contributed directly to parts of the work described in the paper: Ronald Bieber, Hans-Ulrich Block, Michael Dorna, Manfred Gehrke, Johannes Heinecke, Julia Heine, Daniela Kurz, Elsbeth Mastenbroek, Sebastian Millies, Adam Przepiorkowski, Stefanie Schachtl, Michael Schiehlen, Feiyu Xu, and several others.

References

- [Alshawi and Crouch1992] Hiyan Alshawi and Richard Crouch. 1992. Monotonic semantic interpretation. In *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, pages 32–39, Newark, Delaware, June. ACL. Also available as SRI International Technical Report CRC-022, Cambridge, England.
- [Alshawi et al.1991] Hiyan Alshawi, David M. Carter, Björn Gambäck, and Manny Rayner. 1991. Translation by Quasi Logical Form transfer. In *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics*, pages 161–168, University of California, Berkeley, California, July. ACL. Also available as SRI International Technical Report CRC-021, Cambridge, England.
- [Block and Schachtl1992] Hans Ulrich Block and Stefanie Schachtl. 1992. Trace and Unification Grammar. In *Proceedings of the 14th International Conference on Computational Linguistics*, volume 2, pages 658–664, Nantes, France, July. ACL.
- [Bos1995] Johan Bos. 1995. Predicate logic unplugged. In *Proceedings of the 10th Amsterdam Colloquium*, University of Amsterdam, Amsterdam, Holland.
- [Copestake et al.1995] Ann Copestake, Dan Flickinger, Rob Malouf, Susanne Riehemann, and Ivan Sag. 1995. Translation using Minimal Recursion Semantics. Ms., Stanford University, Stanford, California.
- [Egg and Lebeth1995] Markus Egg and Kai Lebeth. 1995. Semantic underspecification and modifier attachment ambiguities. In *Proceedings of the Annual Meeting of the German Linguistic Society*, University of Düsseldorf, Düsseldorf, Germany.
- [Groenendijk and Stokhof1991] Jeroen Groenendijk and Martin Stokhof. 1991. Dynamic Predicate Logic. *Linguistics and Philosophy*, 14:39–100.
- [Kamp and Reyle1993] Hans Kamp and Uwe Reyle. 1993. *From Discourse to Logic: An Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and DRT*. Kluwer, Dordrecht, Holland.
- [Reyle1993] Uwe Reyle. 1993. Dealing with ambiguities by underspecification: Construction, representation and deduction. *Journal of Semantics*, 10:123–179.

[Wahlster1993] Wolfgang Wahlster. 1993. Verbmobil: Translation of face-to-face dialogs. In *Proceedings of the 3rd European Conference on Speech Communication and Technology*, pages 29–38, Berlin, Germany, September.